ÖREBRO UNIVERSITET
INSTITUTIONEN FÖR
NATURVETENSKAP OCH TEKNIK

Datorteknik för civilingenjörer, DT509G

Datorteknik, teori

A001
5 högskolepoäng

Skriftlig tentamen

2023-07-01

Programmering grundkurs, DT509G (A001)

Tillåtna hjälpmedel: penna, radergummi, engelska-svenska ordbok

Instruktioner:

- ***Läs igenom alla frågor noga.***
- **Ange tentamenskoden på svarsdokumentet.**
- **Du kan svara på *Svenska* eller *Engelska*.**
- ***Skriv tydligt* (gäller även för en digital tentamen).**
- **Detta är en individuell examination - alla misstankar om otillåtet samarbete kommer att rapporteras.**
- **Ansvarig lärare finns tillgänglig via telefon fr.o.m. andra skrivtimmen.**
- **Skriv läsligt!**
- **förklara och motivera era svar**

Ansvarig lärare: Pascal Rebreyend, tel: 0702001422

(NOTE: Since the exam is on Saturday, send an SMS and I'll call you back as soon as I can)

För betyg G krävs 50% av total poäng (20 på 40)

(26 poång gav betyg 4, och 33 poäng gav betyg 5)

Lycka till!

Question 1 (3 points)

You have a C code where a struct is declared and in the code you use the function sizeof() to print the size used by a variable of this type struct. You run the code and you get a size x. You add a new element to your struct, your recompile your code and run it. You still get the same size x. Explain why this is possible.

Question 2 (4 points)

The product of a matrix A by a vector x is defined by:

$$Ax = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

$$Ax = \begin{bmatrix} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 \end{bmatrix}$$

If you have a code where you should compute the product of A by a lot of different vectors, how you will store the matrix A in order to be compute the product of A by x as fast as possible. (Note: The matrix A is rather big since its number of lines is huge).

Question 3 (3 points)

- How the decimal 73 is coded as a 8 bits number, using the 2's complement notation?
- Same question with the decimal -193
- Same question for 73 and -105 if the notation is the complement to 1.

Question 4 (4 points):

What will be displayed by the following code:
(We run on an architecture where integer are code on 32 bits and double on 64 bits).

```
#include <stdio.h>
#include <stdlib.h>


void main()
{
  int *a;
  double *b;
  int c,d,i;
  a=calloc(20,sizeof(int));
  b=calloc(20,sizeof(double));
  for (i=0;i<20;i++)
    {
      a[i]=rand()%100;
      b[i]=a[i]/3;
    }
  c=(int)b-(int)a;
  b++;
  a++;
  d=(int)b-(int)a;
  printf("V=%d\n",d-c);
}
```

Question 5 (4 points)

Modern cpu use the pipeline techniques to increase their performances. Explain what the latency is in this context. For which type of code a high latency can be a problem and which ones can performs well with a high latency?

Question 6 (4 points)

The new cpu from the company MyWonderfullCPU use the following floating point representation on 64 bits:

- The first 40 bits are the mantissa
- The next 23 bits are the exponant
- The last bit is the sign bit

In the documentation, they write that the exponant is using the 1’s complement notation but nothing is written on how the mantissa is represented. Do you think it’s an error can you guess which representations can be used and which one should not be used and why?

Question 7 (3 points)

When a cache is used between the memory and the CPU, quickly the cache is full. What is the most common policy used when the cache is full?

Question 8 (2 points).

What is the advantage of using microcode in CPU?

Question 9 (6 points)

By using an ARM-like assembly language, labels, CMP instructions, and the BR (branch) family of instructions, translate the following C code into assembly:
Parameters are passed to the function via R0,R1,R2 and values returned by R3. Your CPU has only 6 registers but you can save value to the stack/memory to addresses A0,A1,A2,... if needed (like SAVE A0,Rx or LOAD Rx,A0 to save to or load from memory).

.....

```
int a,b,c,d,k;
k=0;
for (a=1;a<99;a++)
   {
      c=f1(a);
            d=f2(a);
      while (c!=d)
            {
            c=sqrt(a++);
            k++;
            }
  }
printf(”k=%d\n”,k);
```

Question 10 (3 points)

Why a compiler may reorder the code? Is it useful to do so if a CPU has no pipeline?

Question 11 (2 points)

Why the assembler needs to go through the assembly code at least twice to translate the assembly code into an executable?

Question 12 (2 points)

Describe on an ARM cpu how a call to a function should be prepared and how to deal with ip,lr and pc register with functions and functions calls.